RECEIVED
CENTRAL FAX CENTER
JUN 18 2007

Attorney Docket No.: [40101/01101]

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor(s)	:	McCombe et al.
Serial No.	:	09/738,786
Filing Date	:	December 15, 2000
For	:	System and Method for Managing Client Processes
Group Art Unit:	:	2155
Examiner	:	Bharat Barot
Confirmation No.	:	3798

Mail Stop: Appeal Brief - Patent
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Certificate of Facsimile
I hereby certify that this correspondence is being deposited via facsimile addressed to:

Mail Stop: Appeal Brief-Patents
Commissioner for Patents
Alexandria, VA 22313-1450
(571) 273-8300

By: _____ Date: June 18, 2007
Michael J. Marcin Reg. No. 48,198

TRANSMITTAL

Transmitted herewith please find a Reply Brief in response to the Examiner's Answer mailed on April 16, 2007 for filing in the above-identified application. No fees are believed to be required. The Commissioner is hereby authorized to charge any additional required fees to the **Deposit Account of Fay Kaplun & Marcin, LLP No. 50-1492.** A copy of this paper is enclosed for that purpose.

Respectfully submitted,

Dated: June 18, 2007

By: _____
Michael J. Marcin, Reg. 48,198

Fay Kaplun & Marcin, LLP
150 Broadway, Suite 702
New York, NY 10038
Tel: (212) 619-6000
Fax: (212) 619-0276

LAW OFFICES OF
FAY KAPLUN & MARCIN, LLP
INTELLECTUAL PROPERTY LAW

150 BROADWAY, SUITE 702
NEW YORK, NEW YORK 10038
PHONE: (212) 619-6000
FAX: (212) 619-0276
WWW.FKMIPLAW.COM

FACSIMILE COVER SHEET

FAX NO	:	571-273-8300
TO	:	USPTO Mail Stop: Appeals Brief - Patents
FROM	:	Michael J. Marcin, Esq. of Fay Kaplun & Marcin, LLP
DATE	:	June 18, 2007
SUBJECT	:	U.S. Patent Appln. Serial No. 09/738,786 for *A System and Method for Managing Client Processes* Your Ref: 2000.028 Our Reference: 40101/01101

NUMBER OF PAGES INCLUDING COVER : 27

MESSAGE:

Please see attached.

PATENT
Attorney Docket No.: 40101 - 01101

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:	)	
	)	
Kevin McCombe et al.	)	
	)	
Serial No.: 09/738,786	)	Group Art Unit: 2155
	)	
Filed: December 15, 2000	)	Examiner: Bharat Barot
	)	
For: SYSTEM AND METHOD FOR MANAGING CLIENT PROCESSES	)	**Board of Patent Appeals and Interferences**

Mail Stop: Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

REPLY BRIEF UNDER 37 C.F.R. § 41.41

In response to the Examiner's Answer mailed on April 16, 2007 to the Appeal Brief filed November 1, 2006, and pursuant to 37 C.F.R. § 41.41, Appellants present this reply brief in the above-captioned application.

This is an appeal to the Board of Patent Appeals and Interferences from the Examiner's rejection of claims 1-11 in the final Office Action dated June 8, 2006 as clarified in the Advisory Action mailed August 1, 2006 and further clarified in the Examiner's Answer mailed April 16, 2007. The appealed claims are set forth in the attached Claims Appendix.

1. Grounds of Rejection to be Reviewed on Appeal

I. Whether claims 1-11 are unpatentable under 35 U.S.C. § 103(a) as obvious over U.S. Patent No. 6,470,346 to Morwood (hereinafter the "Morwood patent") in view of U.S. Patent No. 6,385,637 to Peters (hereinafter the "Peters patent").

2. Argument

I. The Rejection of Claims 1-11 Under 35 U.S.C. § 103(a) as Being Obvious Over U.S. Patent No. 6,470,346 to Morwood in view of U.S. Patent No. 6,385,637 to Peters Should Be Reversed.

In the Examiner's Answer, the Examiner maintained that it would have been obvious to one of ordinary skill in the art to incorporate the teaching of the Morwood patent in view of the Peters patent in order to teach "a manager task running at a higher priority than the client task, the manager task queuing the client processes into the client task in priority order, wherein *the manager task kills the client task when a current one of the client processes is not completed within a predetermined time period*," as recited in claims 1. (See Examiner's Answer, pp. 4-6).

The Examiner has cited the Morwood patent as disclosing a "manager task [that] kills the client task based on task (process) priority and by maintaining the task (process) queues, i.e., waiting, running, completed, aborts, and removed." (See Id., p. 5). In the response to the Appellants arguments, the Examiner states that the Morwood patent is relied upon for teaching the limitation of "killing the client task..." (See Id., p. 8).

The Morwood patent describes a method for managing and performing computational tasks, wherein the method enables a requesting client to invoke a computation on a remote server. (See the Morwood patent, col. 1, lines 28-30). Furthermore, the Morwood patent states that a manager process creates a computation object for requests received from the dispatcher and puts the object in the manager's "waiting" queue. (See Id., col. 8, lines 37-48). The manager process determines whether another computation object can be running, the manager process selects the highest priority computation object from the "waiting" queue and causes the computation object to be executed. (See Id.). Accordingly, the manager process controls the number of computation objects that run simultaneously and maintains various queues corresponding to the state of the computational object, i.e., "waiting", "running", "completed", "aborted", and "removed". (See Id., col. 9, lines 9-21). Finally, as the status of the computation objects change, the manager process moves the computation objects to one of the various queues corresponding to the status change. (See Id.).

In contrast to claim 1 of the present invention, and in contrast to the Examiner's argument, the Morwood patent fails to teach or suggest that a "manager task kills [a] client task" based on task priority. As discussed above, the manager process of the Morwood patent simply controls how many computation objects can run at the same time, wherein the computation object having the highest priority is executed when it is determined that another computation object can be executed. The Examiner fails to indicate how selecting a computation object having a higher priority for execution is equivalent or analogous to killing a client process based

on priority. It appears that the Examiner equates the Morwood patent's maintaining of a queue for computation objects having a status of "aborted" anticipates killing a client process. However, as stated in the Morwood patent, "[t]he manager process manages these computations by maintaining them in a set of queues." (See the Morwood patent, col. 9, lines 7-9). The Morwood patent fails to disclose that the manager process kills, or aborts, the computation object. While it is apparent that a computation object of the Morwood patent may attain the status of "aborted," the portions of the Morwood patent cited by the Examiner do not disclose that the manager process causes the computation object status to change to "aborted." The manager process simply places the computation object in the "aborted" queue if the computation object is aborted. Unlike the computation objects in the "waiting" queue, wherein the manager process causes the highest priority computation to execute, the Morwood patent does not teach that the manager process causes any of the computation objects to abort. Thus, the Appellants respectfully disagree with the Examiner's assertion that the Morwood patent teaches that a "manager task kills [a] client task" based on task priority.

Furthermore, the Examiner maintains the assertion that the Peters patent "explicitly discloses that the suspending (killing) a client task when the client processes is not completed within a predetermined time period. (See Examiner's Answer, p. 5). However, as indicated in the Appeal Brief, the Peters patent includes absolutely no disclosure concerning "killing" a client task. Each citation made by the Examiner (*e.g.*, See Peters patent, abstract; Fig. 2; col. 1, ll. 32-53; col. 7, l. 57 - col. 8, l. 8; col. 8, ll. 42-60; col. 9, ll. 5-26; and col. 10, l. 50 -

col. 11, l. 16) (See Examiner's Answer, pp. 5, 7-8) is directed toward the *suspension* of a process, not the *killing* of a process as recited in claim 1. As presented in the Appellants' Appeal Brief, there are several indications within the specification of the Peters patent that teach away from such functionality.

Furthermore, the Appellants further submit evidence to indicate that it is well known in the art that suspending a process is clearly and distinctly separate from killing a process. For instance, the computer language SQL ("Structured Query Language") allows for suspending a process as opposed to killing a process. (Infra Evidence Appendix, Reference 1). Specifically, the command "suspend_process.sql" and "resume_process.sql" are used to suspend and resume an Oracle process, respectively, instead of killing the process. (Infra Id.). Furthermore, the Command Line Process Utility function within Windows NT/2000/XP allows for a process to be viewed, killed, or suspended. (Infra Evidence Appendix, Reference 2). Specifically, a process that is "killed" is terminated without saving files or cleaning up. (Infra Id.). The command for killing a specific process ("-k") using the process name is formatted as: C:\>process -k winword.exe. (Infra Id.). This is clearly a separate command for suspending a process ("-s"). (Infra Id.). "Processes can be suspended if you need some extra CPU cycles without having to kill the process outright." (Infra Id.). The command for suspending a specific process using the process name is formatted as: C:\>process -s winword.exe. (Infra Id.). Finally, it is also important to note that the computer language UNIX also includes separate commands for killing a process and for suspending a process. (Infra Evidence Appendix, Reference 3). In

UNIX, one can kill a process running in the foreground by typing control-c ("^C"). (Infra Id.). Killing a process may become necessary when an executing program is in an infinite loop. (Infra Id.). UNIX also includes a separate command for suspending a process running in the foreground, namely control-z ("^Z"). (Infra Id.). Furthermore, the kill command in UNIX allows for a *suspended* process to be *killed*. (Infra Id.). Since there are two separate command, serving two separate purposes, and since a suspended process may or may not become a killed process, it is clear to any person skilled the in the art of UNIX programming (or SQL programming, or Windows NT/2000/XP programming) that there is a distinction between killing a process and suspending a process. Accordingly, a *suspension* of a process (as disclosed in the Peters patent) entails a restarting of the process at a later time at the point the process was discontinued from executing. In contrast, a *killing* of a process (as recited in claim 1 of the present invention) ends the execution of the process. If the process is chosen to run at a later time, the process is executed anew.

Thus, it is respectfully submitted that neither the Morwood patent nor the Peters patent, either alone or in combination, teaches or suggests "that the manager task kills the client task when a current one of the client processes is not completed within a predetermined time period," as recited in claims 1. Accordingly, it is respectfully submitted that claim 1 is therefore allowable. The Appellants respectfully request that the Board overturn the Examiner's rejection under 35 U.S.C. § 103(a) of claim 1. Because claims 2-5 depend from and, therefore, include all the limitations of claim 1, Appellants respectfully submit that these claims are also allowable and

request that the Examiner's rejections of these claims be overturned.

The Examiner rejected claims 6-10 on the same grounds as claims 1-5, indicating that claims 6-10 were merely a method of operation for the apparatus of claims 1-5. The Examiner used the same rationale to reject claim 11, indicating that claim 11 was merely a computer-readable storage medium storing a set of instructions to manage the apparatus defined in claim 1. For the reasons stated above with respect to claim 1, Appellants respectfully submit that claims 6-11 are allowable and request that the Examiner's rejections of these claims be overturned.

3. Conclusions

For the reasons set forth above, Appellants respectfully request that the Board reverse the final rejections of the claims by the Examiner under 35 U.S.C. § 103(a), and indicate that claims 1-11 are allowable.

Respectfully submitted,

Date: June 18, 2007

By: _____
Michael J. Marcin (Reg. No. 48,198)

Fay Kaplun & Marcin, LLP
150 Broadway, Suite 702
New York, NY 10038
Tel: (212) 619-6000
Fax: (212) 619-0276

RECEIVED
CENTRAL FAX CENTER
JUN 18 2007

Serial No.: 09/738,786
Group Art Unit: 2155
Attorney Docket No.: 40101 - 01101

CLAIMS APPENDIX

1. (Rejected) A system for managing a plurality of client processes, comprising:

a client task within which the client processes will be executed; and

a manager task running at a higher priority than the client task, the manager task queuing the client processes into the client task in priority order, wherein the manager task kills the client task when a current one of the client processes is not completed within a predetermined time period.

2. (Rejected) The system according to claim 1, wherein the manager task restarts the client task and queues a next one of the client processes into the client task.

3. (Rejected) The system according to claim 1, wherein the manager task restarts the client task and requeues the current client process into the client task.

4. (Rejected) The system according to claim 1, wherein the client task sends a response to the manager task indicating the execution of the current client process is complete.

5. (Rejected) The system according to claim 4, wherein the manager task, when receiving the response from the client task, queues a next one of the client processes into the client task.

6. (Rejected) A method for managing a plurality of client processes, comprising the steps of:

queuing a first one of the client processes into a client task, wherein the first client process is executed within the client task; and

killing execution of the client task by a manager task executing at a priority higher than that of the client task when the first client process is not completed within a predetermined time period.

7. (Rejected) The method according to claim 6, further comprising the step of:

releasing a first semaphore by the manager task, wherein the client task does not execute until the first semaphore is released by the manager task.

8. (Rejected) The method according to claim 7, further comprising the step of:

releasing a second semaphore by the client task indicating the execution of the first client process is complete.

9. (Rejected) The method according to claim 6, further comprising the steps of:

restarting the client task by the manager task; and

queuing a second one of the client processes into the client task.

10. (Rejected) The method according to claim 6, further comprising the steps of:

restarting the client task by the manager task; and

requeuing the first client process into the client task

11. (Rejected) A computer-readable storage medium storing a set of instructions, the set of instructions capable of being executed by a processor to manage a plurality of client processes, the set of instructions performing the steps of:

queuing a first one of the client processes into a client task, wherein the first client process is executed within the client task; and

killing execution of the client task by a manager task executing at a priority higher than that of the client task when the first client process is not completed within a predetermined time period.

Serial No.: 09/738,786
Group Art Unit: 2155
Attorney Docket No.: 40101 - 01101

EVIDENCE APPENDIX

Reference 1:

"Don't Kill That Process - Suspend It!", Ixora Pty Ltd,

http://www.ixora.com.au/tips/admin/suspend.htm (Visited on 6/18/2007).

Reference 2:

"Command Line Process Viewer/Killer/Suspender for Windows NT/2000/XP", Craig Peacock, 4/6/2007, http://www.beyondlogic.org/consulting/processutil/processutil.htm (Visited on 6/18/2007).

Reference 3:

"UNIX Tutorial Five", M. Stonebank, 10/9/2000,

http://www.ee.surrey.ac.uk/Teaching/Unix/unix5.html (Visited on 6/18/2007).

Don't Kill That Process - Suspend It!

Have you ever had to kill a long-running batch process because it had overrun its window and was affecting online performance? Well, next time, think again, because there may be a better solution. You may be able to suspend such a process instead of killing it.

If you kill the process it may take longer to roll back than it would have needed to run to completion, and all of the resources used will have been wasted. However, there is an undocumented command that can be used to suspend an Oracle process indefinitely. If you suspend the process, it will consume no more resources until you resume it.

The APT scripts suspend_process.sql and resume_process.sql can be used to suspend and resume Oracle processes. However, don't try to use them under NT, before release 8.1.6, because they suspend the current session instead!

There are, however, two circumstances in which suspending and later resuming a process is not suitable. These are:

1. If other processes need locks that the long-running process is holding in an incompatible mode.
2. If the long-running process is dependent for read consistency on rollback segment information that would be overwritten while it is suspended.

Monday, June 18th, 2007

Command Line Process Viewer/Killer/Suspender for Windows NT/2000/XP

Want a small command line utility to view, kill, suspend or set the priority and affinity of processes, perhaps from a batch file? . . Has a virus disabled your Task Manager? . . or perhaps your Administrator has?

The Command Line Process Utility will function even when the task manager is disabled and/or the dreaded "Task Manager has been disabled by your Administrator" dialog box appears.

Works on remote machines with the Microsoft Telnet Server (tlntsvr) found on Windows 2000 and XP or with BeyondExec for Windows NT4/2000/XP.

View processes, owners, and CPU time . .

```
Command Line Process Viewer/Killer/Suspender for Windows NT/2000/XP V2.01
Copyright(C) 2002-2003 Craig.Peacock@beyondlogic.org

       ImageName   PID Threads Priority CPU%
[System Process]     0       1        0 100 Error 0x6 : The handle is invalid.
          System     8      43        8   0 Error 0x5 : Access is denied.
        SMSS.EXE   180       6       11   0 NT AUTHORITY\SYSTEM
       CSRSS.EXE   204      11       13   0 NT AUTHORITY\SYSTEM
    WINLOGON.EXE   224      16       13   0 NT AUTHORITY\SYSTEM
    SERVICES.EXE   252      33        9   0 NT AUTHORITY\SYSTEM
       LSASS.EXE   264      16        9   0 NT AUTHORITY\SYSTEM
     svchost.exe   436      10        8   0 NT AUTHORITY\SYSTEM
     spoolsv.exe   468      15        8   0 NT AUTHORITY\SYSTEM
    CrypServ.exe   496       3       13   0 NT AUTHORITY\SYSTEM
     svchost.exe   512      28        8   0 NT AUTHORITY\SYSTEM
     hidserv.exe   532       4        8   0 NT AUTHORITY\SYSTEM
  jtagserver.exe   560       3        8   0 NT AUTHORITY\SYSTEM
         mdm.exe   584       6        8   0 NT AUTHORITY\SYSTEM
     nvsvc32.exe   628       2        8   0 NT AUTHORITY\SYSTEM
      regsvc.exe   664       2        8   0 NT AUTHORITY\SYSTEM
      mstask.exe   704       6        8   0 NT AUTHORITY\SYSTEM
      stisvc.exe   728       4        8   0 NT AUTHORITY\SYSTEM
     WinMgmt.exe   804       3        8   0 NT AUTHORITY\SYSTEM
    mspmspsv.exe   876       2        8   0 NT AUTHORITY\SYSTEM
     svchost.exe   896       5        8   0 NT AUTHORITY\SYSTEM
    explorer.exe   616      15        8   0 NEPTUNE\Administrator
       mixer.exe  1092       3        8   0 NEPTUNE\Administrator
    PRISMSTA.exe  1048       1        8   0 NEPTUNE\Administrator
    rundll32.exe   952       2        8   0 NEPTUNE\Administrator
    DIRECTCD.EXE   960       3        8   0 NEPTUNE\Administrator
    internat.exe  1180       1        8   0 NEPTUNE\Administrator
        OSA.EXE  1192       2        8   0 NEPTUNE\Administrator
        Icq.exe  1200      11        8   0 NEPTUNE\Administrator
      devenv.exe  1324       4        8   0 NEPTUNE\Administrator
```

```
IEXPLORE.EXE  1140     7      8    0 NEPTUNE\Administrator
     CMD.EXE  1340     1      8    0 NEPTUNE\Administrator
 Process.exe  1132     1      8    0 NEPTUNE\Administrator
```

Additional switches can be used to display User and Kernel Times (**-t**) or the Creation Time of processes (**-c**).

Kill Processes . . .

Processes can be killed immediately (terminated without saving files or cleaning up) by specifying either the name or the PID (Process IDentifier). In cases where there are multiple processes running with the same name and your desire is to kill a specific process you will need to use the PID.

```
C:\>process -k 748

Command Line Process Viewer/Killer/Suspender for Windows NT/2000/XP V2.01
Copyright(C) 2002-2003 Craig.Peacock@beyondlogic.org
Killing PID 748 'winword.exe'
```

If an image name such as iexplore.exe is specified, the utility will kill all processes by that name.

```
C:\>process -k iexplore.exe

Command Line Process Viewer/Killer/Suspender for Windows NT/2000/XP V2.01
Copyright(C) 2002-2003 Craig.Peacock@beyondlogic.org
Killing PID 996 'iexplore.exe'
Killing PID 1832 'iexplore.exe'
Killing PID 1852 'iexplore.exe'
Killing PID 1692 'iexplore.exe'
```

Close Processes . . .

On the other hand if you want to gracefully close programs by sending them a WM_CLOSE message first, you can used the -q option. This allows processes to clean up, save files, flush buffers etc. However it can cause deadlocks. e.g trying to close Microsoft Word when a unsaved, but edited document is open will generate a dialog box "Do you want to save changes to document 1?". This will prevent winword.exe from exiting until a user responds to the prompt.

```
C:\>process -q wordpad.exe

Command Line Process Viewer/Killer/Suspender for Windows NT/2000/XP V2.01
Copyright(C) 2002-2003 Craig.Peacock@beyondlogic.org
Sending PID 1836 'wordpad.exe' WM_CLOSE Message. Timeout is 60 seconds.
wordpad.exe (PID 1836) has been closed successfully.
```

When this option is used a WM_CLOSE message is immediately sent to the process. It then waits up to a default of 60 seconds for the program to clean up and gracefully close before it is killed. The different timeout can be specified as an option after the PID/Image Name.

Suspend & Resume Processes . . .

Processes can be suspended if you need some extra CPU cycles without having to kill the process outright. Once the requirement for the extra CPU cycles has passed you may resume the process and carry on from where you left off. The process is suspended by sleeping all the processes'

active threads.

```
C:\>process -s winword.exe

Command Line Process Viewer/Killer/Suspender for Windows NT/2000/XP V2.01
Copyright(C) 2002-2003 Craig.Peacock@beyondlogic.org
Suspending PID 748 'winword.exe'
Threads [1084][308]
```

Suspending a process causes the threads to stop executing user-mode (application) code. It also increments a suspend count for each thread. Therefore if a process is suspended twice, two resume operations will be required to resume the process (Decrement the suspend count to zero).

Change the priority of processes . . .

When viewing the list of processes, the 4th column shows the base priority of a process. This is a numeric value from zero (lowest priority) to 31 (highest priority). You may set the base priority of a process by specifying one of the priority classes below.

Low	4
BelowNormal	6
Normal	8
AboveNormal	10
High	13
Realtime	24

Please note Windows NT4 does not support the Above Normal and Below Normal priority classes. Specifying these two parameters on a Windows NT4 machine will result in a " The Parameter is incorrect " error.

```
C:\>process -p winword.exe high

Command Line Process Viewer/Killer/Suspender for Windows NT/2000/XP V2.01
Copyright(C) 2002-2003 Craig.Peacock@beyondlogic.org
Setting PriorityClass on PID 748 'winword.exe' to 128
```

Change the affinity of processes . . .

The affinity is a mask which indicates on which processors (CPUs) a process can run. This is only useful on multiprocessor systems. When the -a option is used in conjunction with a process name or PID, the utility will show the System Affinity Mask and the Process Affinity Mask. The System Affinity Mask shows how many configured processors are currently available in a system. The Process Affinity Mask indicates on what processor(s) the specified process can run on.

```
C:\>process -a wordpad.exe

Command Line Process Viewer/Killer/Suspender for Windows NT/2000/XP V2.01
Copyright(C) 2002-2003 Craig.Peacock@beyondlogic.org
Getting Affinity Mask for PID 1084 'wordpad.exe'
System  : 0x0001 0b00000000000000000000000000000011  [2 Installed Processor(s)]
Process : 0x0001 0b00000000000000000000000000000011
```

To set the affinity mask, simply append the binary mask after the PID/Image Name. Any leading

zeros are ignored, so there is no requirement to enter the full 32 bit mask.

```
C:\>process -a wordpad.exe 01

Command Line Process Viewer/Killer/Suspender for Windows NT/2000/XP V2.01
Copyright(C) 2002-2003 Craig.Peacock@beyondlogic.org
Setting Affinity Mask for PID 1084 'wordpad.exe'
Affinity Mask Successfully Set to 00000000000000000000000000000001
```

Download

- Version 2.03, 25K bytes. (Freeware)

Now supports Windows NT4 Workstation and Server, plus continued support for Windows 2000/XP in a single executable.

Revision History

- 5th June 2003 - Version 2.03
  - Added -c switch which displays the creation times of processes.
- 29th May 2003 - Version 2.02
  - Corrected Inaccurate CPU % Times.
  - Added -t switch which displays both User Mode and Kernel Mode CPU times.
- 15th May 2003 - Version 2.01
  - Fixed memory allocation errors for systems with greater than 100 processes. Application will handle a maximum of 65535 processes.
  - Fixed bug in -q, -k when used with PID. Specifying a PID would kill all processes with the same name than the specified process.
  - Fixed bug with the -a switch when used with PID.
- 26th April 2003 - Version 2.00pre1 (Pre-Release Beta)
  - Caved in to overwhelming demand for support for Windows NT4. Rewrote code to detect operating system and use appropriate API calls plus a couple of undocumented calls to provide all the functionality of previous versions yet across all three NT platforms.
  - Added preliminary support for the setting and display of Affinity Masks for multi processor systems.
  - Added support for killing multiple processes by name. e.g using -k iexplorer.exe will kill all running instances of Internet Explorer, something previously accomplished by a batch file.
  - Added the ability to specify the timeout for the -q option.
  - Improved OpenProcess access so CPU time can now be sought from processes we don't have adequate rights too.
- 15th April 2003 - Version 1.03
  - Modified string to number conversion to correct problem with strings contain leading numbers. eg process -s 3dsmax.exe would try to suspend the process with PID 3 and not 3dsmax.exe.
  - Added -q Send WM_CLOSE message option. This will gracefully Issue a WM_CLOSE message to the program and wait for it to close.
- 21st December 2002 - Version 1.01
  - Corrected problems with exit codes
    - 0 = Success (Process found and desired action performed)
    - 1 = Miscellaneous Error.
    - 2 = Cannot find Process (No processes left my this name)
- 22nd September 2002 - Version 1.00
  - First release to public.

Other Unique and Innovative Software Solutions from Beyond Logic

- Trust-No-Exe - An executable filter for Windows NT/2000/XP
  Allow users to run trusted applications from defined directories, while preventing the execution of non-trusted programs from floppy disk and CDROM drives or from the users e-mail attachment directory. Stop PE viruses in their tracks where on Windows platforms year, nine out of ten of the top viruses were spread via e-mail.

- BeyondExec - Spawn Processes and/or Shutdown Remote Windows NT/2000/XP WorkStations.
  Have you ever wanted to run a process such as an application installer, service pack, virus signature update etc or shutdown a single or group of remote computers without having the burden of installing any remote client on your target computers?

- Bmail - Command Line SMTP Mailer for Batch Jobs
  Bmail is a free but lean command line SMTP mail sender. Bmail allows the user to automate the sending of email messages containing log files, data downloads or error messages on Win32 based computers.

- Delete/Copy by Owner utility for Windows NT/2000/XP
  Have you ever had the need to find, copy or delete files that were owned by a certain user? A great way to back up files created by previous employees or to clean workstations when one leaves.

- PortTalk - A Windows NT/2000/XP I/O Port Device Driver
  A problem that plagues Windows NT/2000/XP, is it's strict control over I/O ports. Unlike Windows 95, 98 or ME, Windows NT/2000/XP will cause an exception (Privileged Instruction) if an attempt is made to access an I/O port that your program is not privileged to access. The PortTalk driver allows existing programs to access selected I/O ports.

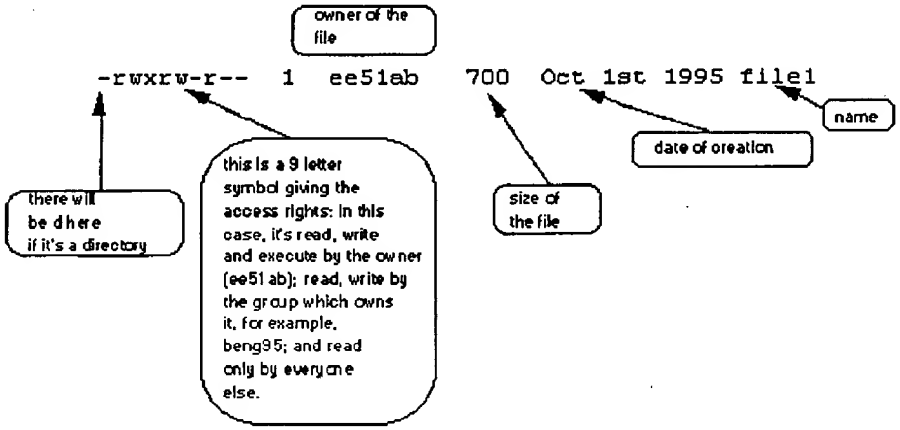
UNIX Tutorial Five

5.1 File system security (access rights)

In your **unixstuff** directory, type

```
% ls -l (l for long listing!)
```

You will see that you now get lots of details about the contents of your directory, similar to the example below.



Each file (and directory) has associated access rights, which may be found by typing `ls -l`. Also, `ls -lg` gives additional information as to which group owns the file (beng95 in the following example):

```
-rwxrw-r-- 1 ee51ab beng95 2450 Sept29 11:52 file1
```

In the left-hand column is a 10 symbol string consisting of the symbols d, r, w, x, -, and, occasionally, s or S. If d is present, it will be at the left hand end of the string, and indicates a directory: otherwise - will be the starting symbol of the string.

The 9 remaining symbols indicate the permissions, or access rights, and are taken as three groups of 3.

- The left group of 3 gives the file permissions for the user that owns the file (or directory) (ee51ab in the above example);
- the middle group gives the permissions for the group of people to whom the file (or directory) belongs (eebeng95 in the above example);
- the rightmost group gives the permissions for all others.

The symbols r, w, etc., have slightly different meanings depending on whether they refer to a simple file or to a directory.

Access rights on files.

- r (or -), indicates read permission (or otherwise), that is, the presence or absence of permission to read and copy the file
- w (or -), indicates write permission (or otherwise), that is, the permission (or otherwise) to change a file
- x (or -), indicates execution permission (or otherwise), that is, the permission to execute a file, where appropriate

Access rights on directories.

- r allows users to list files in the directory;
- w means that users may delete files from the directory or move files into it;
- x means the right to access files in the directory. This implies that you may read files in the directory provided you have read permission on the individual files.

So, in order to read a file, you must have execute permission on the directory containing that file, and hence on any directory containing that directory as a subdirectory, and so on, up the tree.

Some examples

-rwxrwxrwx	a file that everyone can read, write and execute (and delete).
-rw-------	a file that only the owner can read and write - no-one else can read or write and no-one has execution rights (e.g. your mailbox file).

5.2 Changing access rights

chmod (changing a file mode)

Only the owner of a file can use chmod to change the permissions of a file. The options of chmod are as follows

Symbol	Meaning
u	user
g	group
o	other
a	all
r	read
w	write (and delete)
x	execute (and access directory)
+	add permission
-	take away permission

For example, to remove read write and execute permissions on the file **biglist** for the group and others, type

```
% chmod go-rwx biglist
```

This will leave the other permissions unaffected.

To give read and write permissions on the file **biglist** to all,

```
% chmod a+rw biglist
```

Exercise 5a

Try changing access permissions on the file **science.txt** and on the directory **backups**

Use `ls -l` to check that the permissions have changed.

5.3 Processes and Jobs

A process is an executing program identified by a unique PID (process identifier). To see information about your processes, with their associated PID and status, type

```
% ps
```

A process may be in the foreground, in the background, or be suspended. In general the shell does not return the UNIX prompt until the current process has finished executing.

Some processes take a long time to run and hold up the terminal. Backgrounding a long process has the effect that the UNIX prompt is returned immediately, and other tasks can be carried out while the original process continues executing.

Running background processes

To background a process, type an & at the end of the command line. For example, the command `sleep` waits a given number of seconds before continuing. Type

```
% sleep 10
```

This will wait 10 seconds before returning the command prompt %. Until the command prompt is returned, you can do nothing except wait.

To run sleep in the background, type

```
% sleep 10 &
```

```
[1] 6259
```

The & runs the job in the background and returns the prompt straight away, allowing you do run other programs while waiting for that one to finish.

The first line in the above example is typed in by the user; the next line, indicating job number and PID, is returned by the machine. The user is be notified of a job number (numbered from 1) enclosed in square brackets, together with a PID and is notified when a background process is finished. Backgrounding is useful for jobs which will take a long time to complete.

Backgrounding a current foreground process

At the prompt, type

```
% sleep 1000
```

You can suspend the process running in the foreground by typing **^Z**, i.e.hold down the **[Ctrl]** key and type **[z]**. Then to put it in the background, type

```
% bg
```

Note: do not background programs that require user interaction e.g. vi

5.4 Listing suspended and background processes

When a process is running, backgrounded or suspended, it will be entered onto a list along with a job number. To examine this list, type

```
% jobs
```

An example of a job list could be

```
[1] Suspended sleep 1000
[2] Running netscape
[3] Running matlab
```

To restart (foreground) a suspended processes, type

```
% fg %jobnumber
```

For example, to restart sleep 1000, type

```
% fg %1
```

Typing **fg** with no job number foregrounds the last suspended process.

5.5 Killing a process

kill (terminate or signal a process)

It is sometimes necessary to kill a process (for example, when an executing program is in an infinite loop)

To kill a job running in the foreground, type **^C** (control c). For example, run

```
% sleep 100
^C
```

To kill a suspended or background process, type

```
% kill %jobnumber
```

For example, run

```
% sleep 100 &
% jobs
```

If it is job number 4, type

```
% kill %4
```

To check whether this has worked, examine the job list again to see if the process has been removed.

ps (process status)

Alternatively, processes can be killed by finding their process numbers (PIDs) and using kill *PID_number*

```
% sleep 1000 &
% ps
```

```
PID TT S TIME COMMAND
20077 pts/5 S 0:05 sleep 1000
21563 pts/5 T 0:00 netscape
21873 pts/5 S 0:25 nedit
```

To kill off the process **sleep 1000**, type

```
% kill 20077
```

and then type **ps** again to see if it has been removed from the list.

If a process refuses to be killed, uses the **-9** option, i.e. type

```
% kill -9 20077
```

Note: It is not possible to kill off other users' processes !!!

Summary

Command	Meaning

`ls -lag`	list access rights for all files
`chmod [options] file`	change access rights for named file
`command &`	run command in background
`^C`	kill the job running in the foreground
`^Z`	suspend the job running in the foreground
`bg`	background the suspended job
`jobs`	list current jobs
`fg %1`	foreground job number 1
`kill %1`	kill job number 1
`ps`	list current processes
`kill 26152`	kill process number 26152

M.Stonebank@surrey.ac.uk, 9th October 2000

Serial No.: 09/738,786
Group Art Unit: 2155
Attorney Docket No.: 40101 - 01101

RELATED PROCEEDING APPENDIX

No decisions have been rendered regarding the present appeal or any proceedings related thereto.